A knowledge based electronics simulator

Jürgen Herczeg, Michael Herczeg*

Key-words: *electronics simulator, direct manipulation, knowledge representation, expert systems, intelligent tutoring systems, object oriented programming*

Abstract: *This article describes a prototypical computer-based electronics simulator which supports the user during various phases of his task. The system has been built by augmenting a traditional simulator approach by direct manipulation techniques and by representing substantial knowledge of the task in an expert system kernel. The direct manipulation interface allows the user to manipulate the familiar objects of an electronics laboratory on the computer screen. The expert component provides adequate simulation parameters, answers questions about the problem domain and gives examples and partial solutions of the problem at hand. It therefore serves as a kind of intelligent tutor.*

Ein wissensbasierter Elektroniksimulator

Stichworte: *Elektroniksimulator, direkte Manipulation, Wissensrepräsentation, Expertensystem, intelligenter Tutor, objektorientierte Programmierung.*

Zusammenfassung: *Dieser Beitrag beschreibt ein prototypisches Elektroniksimulationssystem, das den Benutzer während verschiedener Phasen der Arbeit unterstützt. Zu diesem Zweck wurde ein herkömmlicher Simulatoransatz durch die Möglichkeit der direkten Manipulation der Arbeitsobjekte auf dem Bildschirm und durch die Repräsentation von Wissen über die Aufgabe in einem Expertensystemteil erweitert. Direkte Manipulation erlaubt dem Benutzer, die vertrauten Objekte eines Elektroniklabors auf dem Bildschirm zu selektieren, zu positionieren und in ihren Eigenschaften zu modifizieren. Der Expertenteil liefert geeignete Simulationsparameter, beantwortet Fragen zum Anwendungsbereich und gibt Beispiele und Teillösungen zum vorliegenden Problem. Er dient demgemäß als intelligenter Tutor.*

* Jürgen Herczeg, Dr. Michael Herczeg, Forschungsgruppe INFORM, Institut für Informatik, Universität Stuttgart, Herdweg 51, D-7000 Stuttgart 1

[1]) "he" is used to represent the 3rd person singular; "she" could equally well be used

1 Introduction

Basically two quite different types of computer systems dominate in today's offices: Most common are text- and form-oriented business application systems; they may be characterized as being monotonous in their appearance and inflexible to use. Then there are highly functional computer aided design systems. While giving the users a feeling of directness of interaction with the problem domain, they put the burden of high complexity on them.

As an intermediate form between these two types of systems, desktop oriented systems have emerged. These systems may be characterized by being highly interactive and giving the user a feeling of directness in interaction with the business application domain, since they model the familiar business environment graphically on the screen and enable their users to manipulate the screen objects via a pointing device (e.g. a mouse). In contrast to the computer aided design systems they reduce complexity by presenting objects and manipulation functions the users already know from their office working environment.

This *desktop metaphor* has been developed within traditional data and text processing tasks. In this paper we show how this apporach may be exploited for a technical laboratory environment, for an electronics lab (called ELAB). This results in an easy to handle technical application system still showing some of the problems of a complex application domain. We will cope with these problems by augmenting the direct manipulation laboratory by an expert component (called ELEX), actively supporting the user in his[1]) task and passively giving advice about the application concepts, whenever the user asks for.

The system has been implemented in an object-oriented, LISP-based knowledge representation language on a VAX 11/780 with high resolution bitmap terminals. We were able to use software tools, like an already existing user interface construction kit based on the window system environment WLISP of the research group INFORM [1], [3].

2 ELAB – A Direct Manipulation Electronics Simulator

With ELAB, the analysis of electrical circuits may be performed on a computer in basically the same way as in a real laboratory. The user – e.g. an electronics expert –

0013-5704/88/5 0220–07 $ 03.00/0

sees an experimentation field with most of the familiar working objects of an electronics laboratory (Figure 1):

- electrical devices (resistors, capacitors, coils)
- voltage source (function generator)
- measuring instruments (oscilloscope, frequency analyser, analog and digital gauges)

The user creates a circuit by means of *direct manipulation* [10], [6]. The circuit may then be "run" by a simulation component. The results of the simulation can be analysed using various instruments. The user modifies the circuit until it shows a satisfying behaviour. To accomplish this task it is not necessary to learn a formal language (like in the SPICE system [12]), a complex computer-aided design system or a graphics package. The user solely learns to select, move, connect, and modify objects on the screen. The desktop metaphor has been transformed into a *lab metaphor*: the visualization of a laboratory on a computer screen.

ELAB is restricted to a small application domain, a micro-world: The system allows the combination of elementary four-poles consisting of resistors, capacitors or coils into more complex circuits (Figure 2). However, the resulting circuits can get so complex that even a human electronics expert is unable to predict their behaviour qualitatively, let alone quantitatively. Some of the most interesting circuits are voltage dividers, filters (e.g. high-pass, low-pass, bandpass) and several resonant circuits. Diverse input signals and initial states of the devices increase the complexity drastically. The voltage source provides periodical signals and pulses in the shape of a sine-curve, a rectangle, a triangle, or a constant. Various instruments allow to analyse the behaviour of the circuit; they look like hardware instruments of a real laboratory, but they have some additional significant advantages. The oscilloscope, for example, is able to protocol simultaneously as many different signals as the user wishes to see. Its coordinate system is variable and it displays not only periodical signals

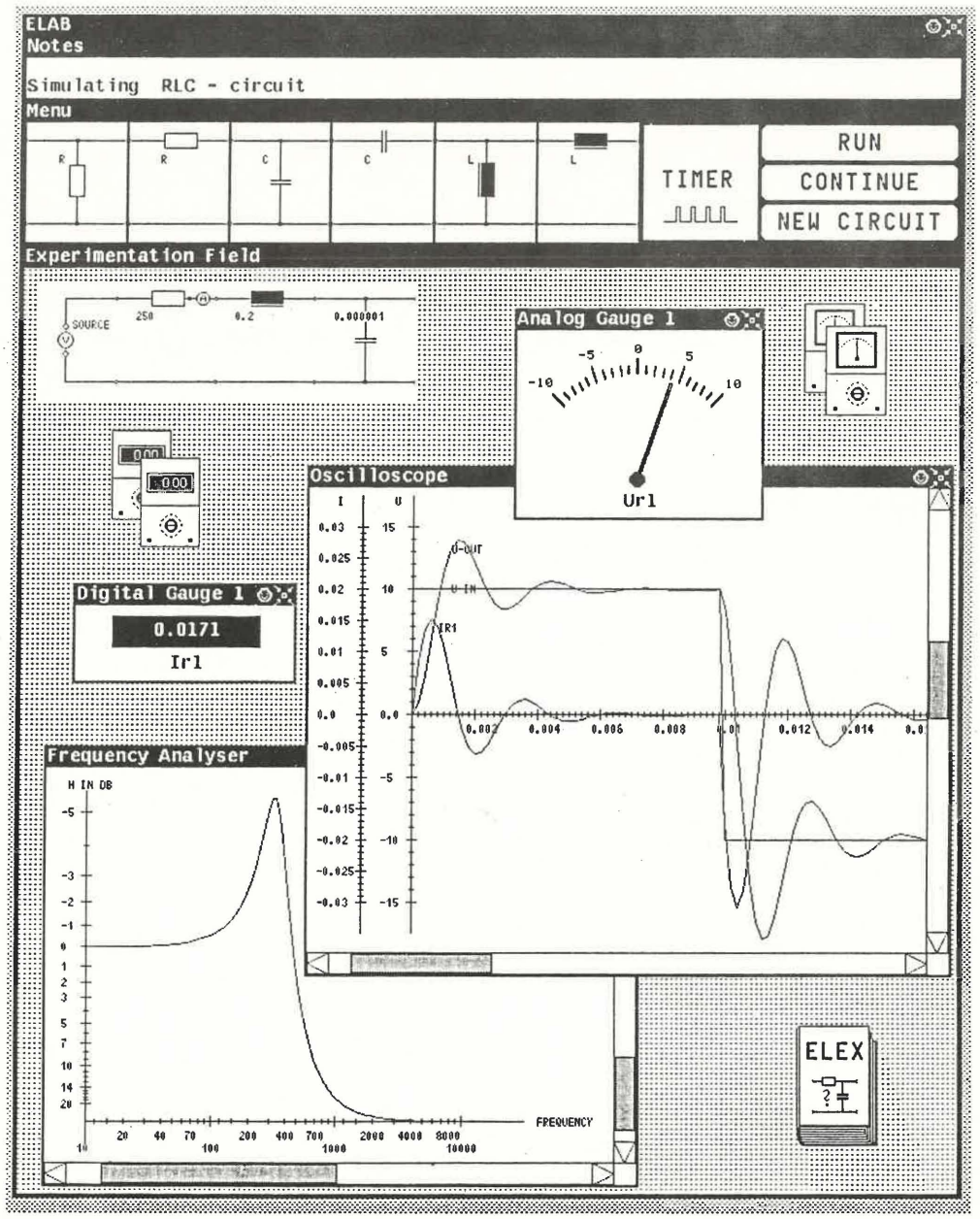

Figure 1
The experimentation field of ELAB

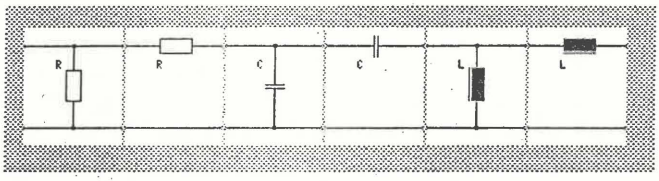

Figure 2 The elementary four-poles of ELAB

like most hardware oscilloscopes do, but also arbitrarily short pulses and varying oscillations. The analog gauges have interactively modifiable scales, hands and labels.

The user plugs the elementary four-poles to a circuit. This is performed with the mouse as a pointing device. *Pop-up-sheets* are used to change the attributes of the four-poles, *softbuttons* to start or continue the simulation. Before or after the simulation, the instruments may be connected to the circuit to measure voltage and current of the devices. While analog and digital gauges show momentary values, the oscilloscope displays the results sampled over the simulation time. The frequency analyser shows the statically defined transfer function of the circuit, i.e. the input/output signal ratio for an arbitrary frequency interval.

The keyboard is solely used for the input of values or when communicating with the expert component ELEX. ELEX

- "watches" the user's actions;
- controls simulation parameters;
- adjusts the coordinate system of the oscilloscope in order to provide a good overview of the sampled information;
- simplifies complicated circuits to simpler ones with the same behaviour;
- builds circuits and adjusts the parameters of the devices according to the user's high level descriptions (e.g. the user specifies that the wants to build a resonant circuit with some predefined resonant frequency);
- explains electronics concepts and enables the user to browse through a network of concepts (e.g. the user asks how to build a low-pass);
- explains what would happen if a device would be modified (e.g. what will happen if the resistance in a band-pass will be increased), and
- tells about characteristic values in a circuit and describes how they are effected by the devices.

3 The Components of the System

ELAB is composed of seven functional components (Figure 3). These components are only conceptually defined, not all of them are reflected in the implementation. ELAB has been implemented in the object-oriented language ObjTalk [9], [8]. Nearly every object which may be recognized on the display by the user is represented as an internal object. A resistor four-pole, for example, is represented by an internal object with the following attributes:

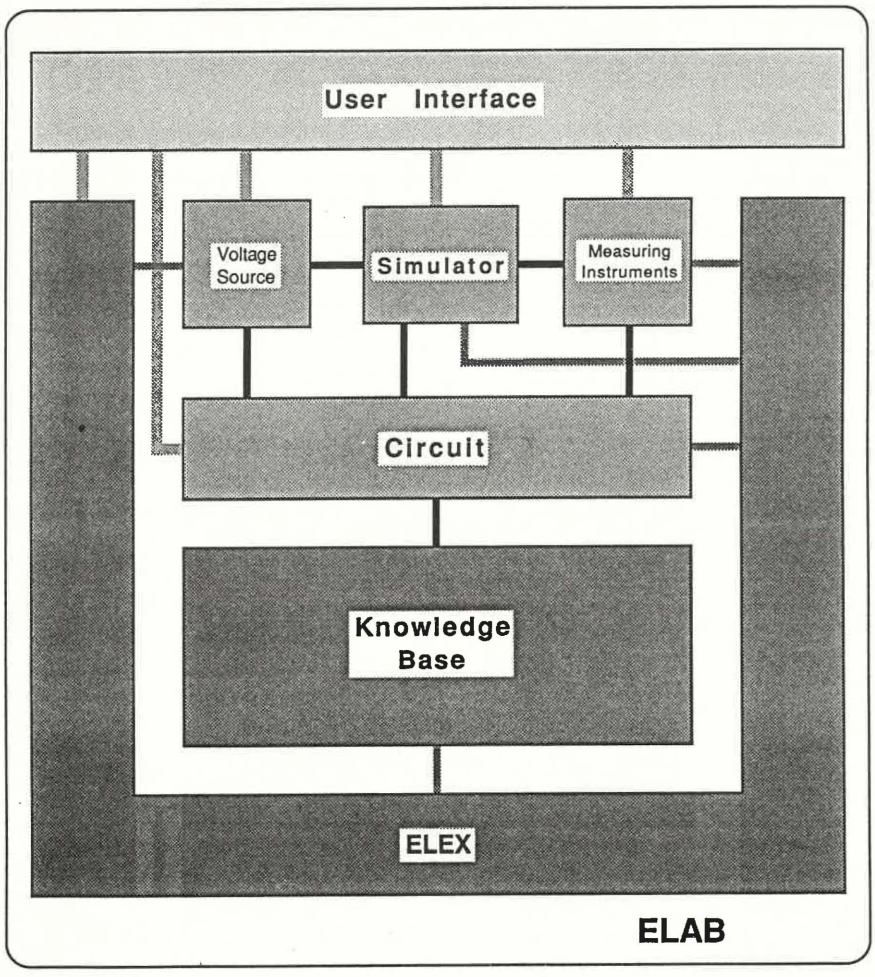

Figure 3
The components of ELAB

Type = resistor	the four-pole consists of a resistor
Orientation = parallel	the resistor is arranged in parallel
Quantity = R	the characteristic quantity is resistance R
R = 1000	resistance in Ohms
U = 10	voltage at the resistor in Volts
I = 0.01	current through the resistor in Amperes
Position = 3	the four-pole is the 3rd four-pole of the circuit
Relative-Position = 2	the four-pole is the 2nd resistor of the circuit
Circuit = <some-circuit>	the circuit which the four-pole belongs to
Icon = <some-object>	icon to visualize the four-pole
Sheet = <some-sheet>	sheet to modify object attributes of the four-pole
U-to-be-Checked? = t	voltage at the resistor is checked by the oscilloscope
I-to-be-Checked? = nil	current through the resistor is not checked

Besides its attributes, an object may have methods that represent its behaviour. An instrument object, for example, has methods to connect or disconnect itself to or from a circuit object. The methods of an object may be invoked by another object by sending a message to it.

The basic principle, how the whole system works, is: *objects are sending message to other objects to invoke methods that change their states or send new messages*. The lines in Figure 3 show the communication paths between objects of different components. Every component consists of a set of objects, communicating with objects inside and outside of the component. The reason for identifying components is to group together objects which perform some definite task in the system. The whole system forms a large network of communicating objects.

The functional components of ELAB are:

circuit:	object that represents the circuit constructed by the user
voltage source:	object that drives the circuit by a user-selected input signal
measuring instruments:	objects that may be connected to the circuit to show its behaviour during and after the simulation
simulator:	a timer generating "time-ticks" to make voltage source, circuit, and instrument objects proceed in time and produce the simulation data
knowledge base:	objects representing electronics concepts and rules to create the circuit object out of the user's graphical specifications, to provide the simulation knowledge for each circuit, and to generate explanations requested via ELEX by the user
ELEX:	objects representing the expert component; ELEX is communicating with all other components
user interface:	user interface objects displaying the internal objects of the other components on the screen and processing input events invoked via mouse and keyboard by the user

4 ELEX – A Simulation and Electronics Expert

To support the ELAB user in solving problems with the laboratory environment, the expert component ELEX has been constructed. It can be divided into three major functional parts:

- a *simulation expert*, managing the access to the circuit knowledge base and setting up the simulation parameters,
- a *laboratory expert*, serving as a tutor which watches the user at work with the ELAB environment, and
- an on-call *electronics expert*, providing different kinds of information about the application domain.

ELEX thus combines the aspects of intelligent tutoring systems [11] and expert systems.

4.1 The Simulation Component

In ELAB a circuit is specified by a sequence of elementary four-poles. This four-pole structure is transformed into a normalized form. Actually, the four-poles are just reordered internally according to a set of precedence rules, maintaining the behaviour of the circuit represented by the four-poles. Using this normalized four-pole structure, a simple pattern-matcher determines a specific object from the ELAB knowledge base which describes the simulation behaviour of the corresponding circuit. This means that only those circuits can be simulated which can be transformed into a circuit contained in the knowledge base. The knowledge base, of course, can be extended by adding the description of any circuit that can be built from the elementary four-poles.

The simulation model is based on a numerical process, relying on the explicitly described behaviour of each circuit in a given time interval when driven by a constant input voltage. This behaviour is specified completely by the change of those electrical quantities which determine the energy in the circuit, i.e. the capacitor voltages and coil currents, and by the interdependencies of all other quantities (device voltages and currents) on those energy-determining quantities. The behaviour of the circuit, of course, depends upon the input voltage, the device parameters and the state of the circuit, i.e. the previous values of the energy-determining quantities. This process is iterated to get the overall behaviour of the circuit when driven by an arbitrary voltage signal. The constant input voltage for each step of the iteration is obtained from the present values of the input signal scanned at constant time-intervals. After each step, the newly calculated values for voltages and currents are propagated to the measuring instruments. The whole simulation process is controlled by one central object which synchronizes the communication between the source, the circuit, and the measuring instruments.

There are a few parameters to be adjusted for each simulation process, such as the time interval length between each iteration and the number of iterations to be carried out in order to get a usable simulation-result. Since the ELAB user is usually not concerned with the simulation model and does not know the behaviour of the circuit in advance, he is not able to adjust these parameters properly and is therefore relieved from this work by the expert. To optimize the simulation parameters the expert utilizes knowledge about the input signal, the circuit behaviour, and possible requirements on the quality of the simulation stated by the user.

4.2 The Laboratory Expert

The laboratory expert supports the ELAB user while setting up experiments. It ensures that the system is always in a consistent and reliable state. It also directs the attention of the user to critical situations that are obviously not intended. Since in most cases they stem from ignorance or negligence on the part of the user, they can only be overcome by an active component, which watches the interactions and makes the user aware of his mistakes and of critical situations. While mistakes are simply undone, in critical situations the expert gives the user the opportunity to undo an action which was probably not intended.

A typical mistake is as follows: the user provides a parameter with a meaningless value, like a negative value for a time period. ELEX rejects the value and prints the expected value range of the parameter.

A critical situation would be the following: the user starts the simulation of a circuit without any connected measuring instrument; no simulation data, except for the final state of the circuit after the simulation, would be inspectable. In this case the expert proposes to abort the simulation, but leaves the decision to the user.

To detect these kinds of problems, continuous communication is required between the expert and the experimentation objects, e.g. the measuring instruments. The expert applies heuristics to decide whether and when the actions of the user are to be interrupted. This is a characteristic problem of active help systems and intelligent tutoring systems.

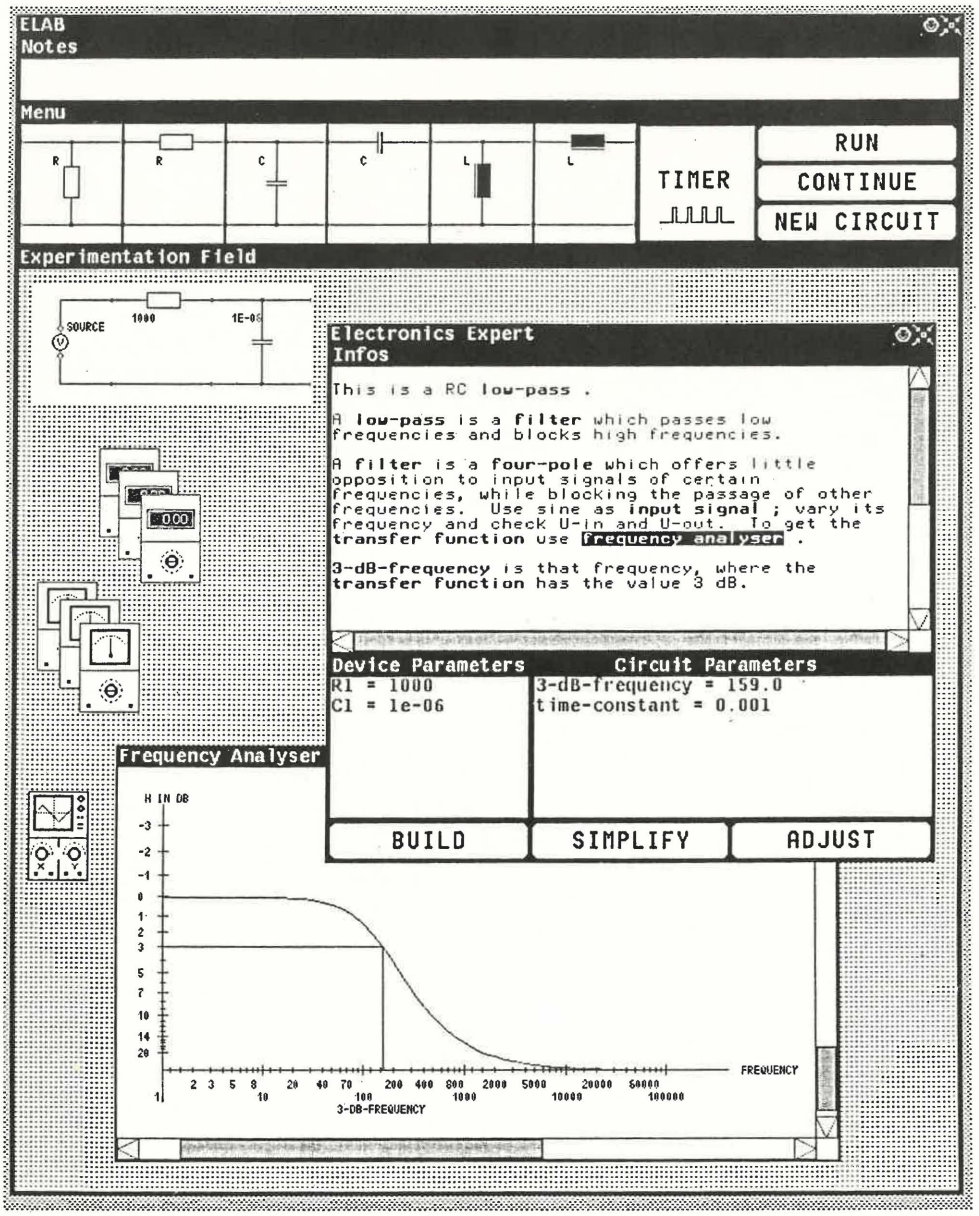

Figure 4
The electronics expert

4.3 The Electronics Expert

The electronics expert adds some features to the experimentation environment, such as providing explanations about various electronics concepts or about specific electrical circuits either built by the user or by the expert. This part of the expert is a key component for a knowledgeable experimentation environment. It enables the user to solve problems by communication on a higher level of the application domain.

The dialog between the user and the expert takes place in a window containing three subwindows – one for textual information, another for the characteristic parameters of the electrical devices and a third for specific parameters of the circuit, e.g. the 3dB-frequency of a low-pass filter – and some softbuttons to trigger specific actions of the expert (Figure 4). The information-window presents textual output of the expert, such as:

- a description of the present circuit,
- information on how to simplify the present circuit,
- explanations of electronics concepts, e.g. low-pass or the 3dB-frequency, and
- statements about the qualitative dependencies between the device parameters and the specific circuit parameters, e.g. how the resistance R1 of the present circuit affects the circuit parameters, or how the 3dB-frequency is affected by the device parameters.

Each piece of information has to be requested by the user, i.e. the expert does not present all available information; it only shows what is particularly interesting for the user. Acutally, information is partly generated at the moment it is requested, by making inferences from general knowledge and specific circuit data.

To request information, the user points to the item in the expert window in which he is interested. Such sensitive items are printed in bold face and are explained when selected by the mouse. In addition, explanations may be augmented by some visual feedback for a better understanding of the explained concept, e.g. the 3dB-frequency is drawn into the diagram of the transfer function (see Figure 4).

Dependencies of one parameter on others may be requested by selecting a parameter and choosing some specification from a menu (Figure 5). This qualitative information can be particularly useful to understand how a circuit works, and how its behaviour is influenced by the parameters of its components.

The electronics expert is able to detect redundancy in the circuit, e.g. two resistors arranged in series can be substituted by one resistor. It is also able to eliminate this redundancy by simplifying the circuit, retaining the overall circuit behaviour, and explaining to the user what has been done. This is particularly useful if a circuit is not contained in the knowledge base but might be reduced to one that can be simulated.

ELEX also helps the user to build special circuits, and to adjust the device parameters so that the circuit shows a particular behaviour, e.g. how to build a low-pass with a 3dB-frequency of 4 kHz. Circuits which can be built automatically by the expert are chosen from menus. To get a

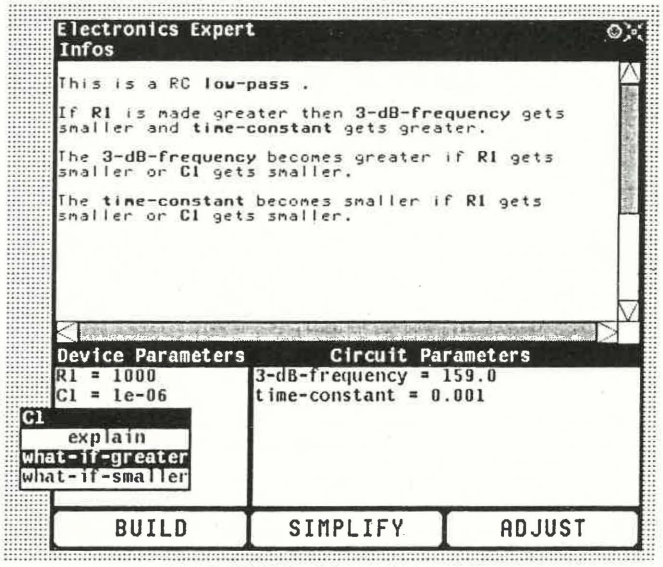

Figure 5 Asking for parameter dependencies

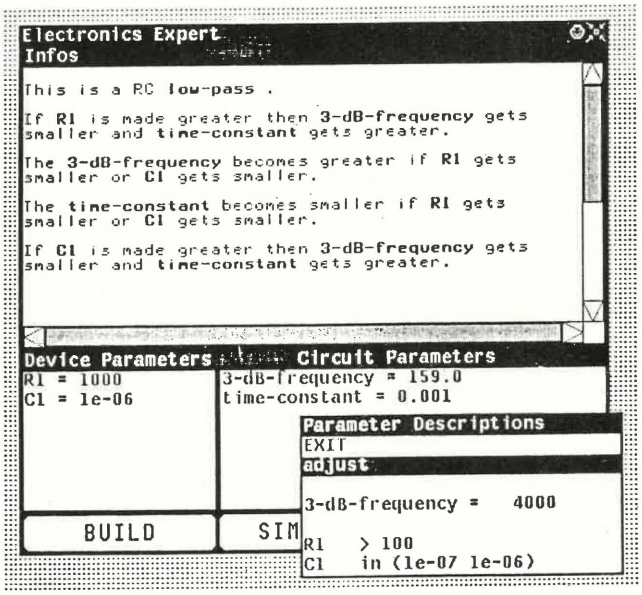

Figure 6 Adjusting the device parameters according to a fixed 3dB-frequency

particular circuit behaviour, a circuit parameter has to be selected and certain parameter constraints to be satisfied by the expert may be specified in a special sheet (Figure 6).

To cope with all aspects described above, the electronics expert is based on a fair amount of explicitly represented *declarative knowledge* as well as *procedural knowledge.*

The *declarative knowledge*, contained in the ELAB knowledge base, includes a hierarchy of circuit concepts and an unstructured collection of arbitrary electronics concept descriptions. Each concept is represented as an object and consists of a description-part and a feedback-part. The description is a text that explains the concept and may refer to other concepts. Thus many concepts may be connected implicitly. The feedback-part describes possible visual feedback actions as mentioned above.

The circuit concepts are explicitly connected by pointers (sub-circuit pointers and super-circuit pointers) to represent the natural concept-subconcept relation. The concept *filter*, for example, has a sub-circuit pointer to the concept *low-pass*, whereas *low-pass*, on the other hand, has a back-pointing super-circuit pointer to *filter*. Each circuit concept may have a set of associated circuit parameters which characterize the concept precisely, for example, *low-pass* has the associated parameter *3dB-frequency*.

The terminal nodes of this hierarchical circuit network are those circuits of the knowledge base which determine the simulation behaviour. They have a pointer to their immediate super-circuits in the hierarchy, e. g. *RC-low-pass* has a super-circuit pointer to *low-pass*, and they also have an explicit description of how to compute specific circuit parameters from the device parameters. All circuit parameters are found starting at the circuit in question, following the super-circuit path through the net. This corresponds to the concept of inheritance in hierarchical networks. The declarative knowledge of ELEX may be enlarged by adding arbitrary new concepts to the knowledge base.

The *procedural knowledge* consists of methods utilizing declarative knowledge to infer new kind of declarative knowledge. This knowledge can either be quantitative or qualitative, and in some cases has to be transformed into textual information for the user.

To eliminate redundancy of a circuit, simplification rules are applied which substitute or remove elementary four-poles in the normalized four-pole structure supplied by the simulation expert.

A numerical evaluation is executed to determine dependencies between device parameters and circuit parameters. To find out how the parameter of a particular device affects a specific circuit parameter ELEX conducts an experiment; it simply changes the value for the device parameter, watches the effect on the circuit parameter, and determines the qualitative dependency, which is passed on the the user.

A somewhat more difficult method is used to adjust the device parameters according to a set of constraints specified by the user (see also Figure 6). A *relaxation* method based on a numerical, iterative process is employed to find the roots of an equation with several variables [7]. The equation is derived from the formula that describes how a specific circuit parameter is related to the device parameters.

The electronics expert thus presents a way to derive both quantitative and qualitative knowledge by combining symbolic manipulation and numerical evaluation methods.

5 Conclusions

In the domain of electronics simulation, traditional, batch-oriented systems with inadequate user interfaces are still predominant. Interactive systems using direct manipulation, however, represent the problem domain in a much more natural way. We have built the prototypical system ELAB to demonstrate how to extend the desktop metaphor to a *lab metaphor* for technical applications, such as electronics simulation.

The expert component ELEX has been added to the laboratory environment to support the user in performing experiments and solving problems with ELAB. To cope with these tasks, both knowledge about the application domain and the particular simulated laboratory environment have been represented. This proved to be useful to put the finishing touches on a system based on the visualization of a complex application domain.

ELAB has been implemented in the object-oriented language ObjTalk. This turned out to be very suitable for this approach for several reasons: Object-oriented programming makes it easy to describe the communication between various instances of the problem domain in a quite natural way, and, on the other hand, supports representation of knowledge. Further, the user interface could be easily constructed using an object-oriented construction kit based on the window system WLISP.

In ELAB the aspects of intelligent tutoring systems and expert systems have been combined in order to make the system suitable for both novices and advanced users as a learning environment. To avoid the present restriction of the simulator to predefined electrical circuits and the prevailing application of circuit-specific knowledge a general purpose simulator, such as SPICE, could be used.

References

[1] *H.-D. Böcker, F. Fabian, Jr., A. C. Lemke:* WLisp: A Window Based Programming Environment for FranzLisp. In Proceedings of the First Pan Pacific Computer Conference, Volume 1, pp. 580–595. The Australian Computer Society, Melbourne, Australia, September, 1985.

[2] *G. Fischer, R. Gunzenhäuser* (Herausgeber): Methoden und Werkzeuge zur Gestaltung benutzergerechter Computersysteme. Verlag Walter de Gruyter & Co., Berlin – New York, 1986.

[3] *M. Herczeg:* INFORM-Manual: Icons, Version 3.12. Computer Science Institute, University of Stuttgart, 1986, Translated by V. M. Patten, February 1986.

[4] *M. Herczeg:* Eine objektorientierte Architektur für wissensbasierte Benutzerschnittstellen. Dissertation, Fakultät Mathematik und Informatik der Universität Stuttgart, Dezember 1986.

[5] *M. Herczeg, D. Maier, C. Rathke, W.-F. Riekert:* Vom Dialogsystem zur wissensbasierten Mensch-Computer-Kommunikation. In ONLINE '85. 8. Europäische Kongreßmesse für Technische Kommunikation, pp. 2P–2P–14. Düsseldorf, Februar, 1985.

[6] *E. L. Hutchins, J. D. Hollan, D. A. Norman:* Direct Manipulation Interfaces. In D. A. Norman, S. Draper (Editors), User Centered System Design: New Perspectives on Human-Computer Interaction. Lawrence Erlbaum Associates Ltd., 1986.

[7] *M. Konopasek, S. Jayaraman:* Expert Systems for Personal Computers. The TK! Solver Approach. In Byte, pp. 137–156, Mai, 1984.

[8] *A. Lemke:* ObjTalk 84 Reference Manual. Technical Report CU-CS-291-85, University of Colorado, Boulder, 1985.

[9] *C. Rathke:* ObjTalk. Repräsentation von Wissen in einer objektorientierten Sprache. Dissertation, Fakultät Mathematik und Informatik der Universität Stuttgart, Oktober 1986.

[10] *B. Shneiderman:* Direct Manipulation: A Step Beyond Programming Languages. In IEEE Computer 16 (8), pp. 57–69, August, 1983.

[11] *D. Sleeman, J. S. Brown* (editors): Intelligent Tutoring Systems. Computers and People Series. Academic Press, London – New York, 1982.

[12] *A. Vladimirescu, A. R. Newton, D. O. Pederson:* SPICE 2G.0 User's Guide. Technical Report, University of California, Berkeley, 1980.

Angewandte Informatik